Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Original) A method for providing random bits comprising:

storing random bits in a buffer;

retrieving a quantity of random bits from the buffer at a source location;

altering the random bits in the buffer at the source location;

generating a new quantity of random bits based on the retrieved quantity of random bits;

advancing the source location to the next location in the buffer; and

placing the source location to the beginning of the buffer plus an offset when the next source location is beyond the limit of the buffer.

2. (Original) The method of claim 1 wherein storing random bits comprises:

storing a quantity of random bits in a buffer at a destination location;

advancing the destination location to the next destination location in the buffer; and

placing the destination location at the beginning of the buffer plus an offset when the next destination location is beyond the limit of the buffer.

3. (Original) The method of claim 1 wherein storing random bits comprises:

organizing the buffer into blocks;

storing a quantity of random bits in a first block at a chosen destination location;

advancing the destination location to a chosen destination location in a different block in the

buffer; and

placing the destination location to a chosen destination location in a different block to the beginning of the buffer plus an offset when the next destination location is beyond the limit to the buffer.

4. (Original) The method of claim 1 wherein storing random bits comprises:

organizing the buffer into blocks;

storing a quantity of random bits in a first block at a chosen destination location; and

advancing the destination location to a destination location in a different chosen block in the buffer wherein the chosen block is chosen with preference for a least recently updated block.

5. (Original) The method of claim 1 wherein altering the random bits in the buffer at the source location comprises incrementing a digital value comprised of the quantity of random bits at the source location.

6. (Original) The method of claim 1 wherein generating a new quantity of random bits comprises applying a one-way hash function to the selected first quantity of random bits.

7. (Original) An apparatus for providing random bits comprising:

buffer;

input manager that is capable of receiving random bits and storing said random bits in the buffer; and

output manager that is capable of:

retrieving a quantity of random bits from the buffer at a source location;

altering the random bits in the buffer at the source location;

generating a new first quantity of bits based on the retrieved quantity of random bits;

advancing the source location to a next location in the buffer; and

placing the source location to the beginning of the buffer plus an offset when the next source location is beyond the limit of the buffer.

8. (Original) The apparatus of claim 7 wherein the input manager comprises:

input register that is capable of holding a quantity of random bits;

destination pointer that is capable of generating a write address;

incrementer that is capable of incrementing a value stored in the destination pointer; and

input controller that is capable of:

loading random bits into the input register;

issuing a write signal to the buffer when the input register is holding random bits;

commanding the incrementer to increment the destination pointer after the write signal is de-asserted.

9. (Original) The apparatus of claim 7 wherein the input manager comprises:

input register that is capable of holding a quantity of random bits;

destination pointer that is capable of generating a write address that includes a block identification field;

incrementer that is capable of incrementing the block identification field; and

input controller that is capable of:

loading random bits into the input register;

issuing a write signal to the buffer when the input register is holding random bits;

commanding the incrementer to increment the block identification field of the destination pointer after the write signal is de-asserted.

10. (Original) The apparatus of claim 7 wherein the input manager comprises:

input register that is capable of holding a quantity of random bits;

destination pointer that is capable of generating a write address that includes a block identification field;

block selector that is capable of loading the block identification field with an indicator according to a least recently updated block; and

input controller that is capable of:

loading random bits into the input register;

issuing a write signal to the buffer when the input register is holding random bits;

commanding the block selector to update the block identification field of the destination pointer after the write signal is de-asserted.

11. (Original) The apparatus of claim 7 wherein the output manager comprises:

output register for holding random bits received from the buffer;

value incrementer capable of incrementing the numeric value of bits held in the output register; and

output controller capable of issuing a write signal to the buffer when an output of the value incrementer is available at said buffer.

12. (Original) The apparatus of claim 7 wherein the output manager comprises: output register for holding random bits received from the buffer;

transformation table that generates an output based on an input received from the output register and wherein the transformation table is loaded with a one-way hash function; and

output controller that issues a capture signal to the output register that causes the output register to store random bits.

13. (Original) A random bit provisioning unit comprising: processor for executing instructions;

memory; and

instruction sequences stored in the memory including: input process instruction sequence that, when executed by the processor, minimally causes the processor to store random bits in a buffer region in the memory; and

output process instruction sequence that, when executed by the processor, minimally causes the processor to:

retrieve a quantity of random bits from the buffer region in the memory at a source location;

alter the random bits at the source location;

generate a new quantity of random bits based on the retrieved quantity of random bits;

advance the source location to the next location in the buffer region; and

place the source location to the beginning of the buffer region plus an offset when the next location is beyond the limit of the buffer region.

14. (Original) The random bit provisioning unit of claim 13 wherein the input process instruction sequence includes a write module instruction sequence that causes the processor to store random bits by minimally causing the processor to:

store a quantity of random bits in the buffer region at a destination address;

advance the destination address to the next location in the buffer region;

place the destination address to the beginning of the buffer region plus an offset when the next destination address is beyond the range of the buffer region.

15. (Original) The random bit provisioning unit of claim 13 wherein the input process instruction sequence includes a write module instruction sequence that causes the processor to store random bits by minimally causing the processor to:

store a quantity of random bits in the buffer region at a destination address;

advance the destination address to the next location in different block in the buffer region;

place the destination address to the beginning of the buffer region plus an offset when the next destination address is beyond the range of the buffer region.

16. (Original) The random bit provisioning unit of claim 13 wherein the input process instruction sequence includes a write module instruction sequence that causes the processor to store random bits by minimally causing the processor to:

store a second quantity of random bits in the buffer region at a destination address; and

advance the destination address to the next location in different block in the buffer region

wherein the different block is chosen according to a least updated block.

17. (Original) The random bit provisioning unit of claim 13 wherein the output process instruction sequence includes an increment instruction sequence that causes the processor to alter the random bits at the source location by minimally causing the processor to increment in the buffer region in the memory a digital value representing the retrieved quantity of random bits.

18. (Original) The random bit provisioning unit of claim 13 wherein the output process instruction sequence includes a transform function that causes the processor to generate a new quantity of random bits by minimally causing the processor to apply a one-way hash function to the retrieved quantity of random bits.

19. (Original) A computer-readable medium having computer-executable functions for providing a random bit stream comprising:

input process instruction sequence that, when executed by a processor, minimally causes the processor to store random bits in a buffer region in the memory; and

output process instruction sequence that, when executed by a processor, minimally causes the processor to: retrieve a quantity of random bits from the memory at a source location;

alter the random bits at the source location;

generate a new quantity of random bits based on the retrieved first quantity of random bits;

advance the source location to the next location in the buffer region; and

place the source location to the beginning of the buffer region plus an offset when the next location is beyond the limit of the buffer region.

20. (Original) The computer-readable medium of claim 13 wherein the input process instruction sequence includes a write module instruction sequence that causes the processor to store random bits by minimally causing the processor to:

store a quantity of random bits in the buffer region at a destination address;

advance the destination address to the next location in the buffer region;

place the destination address to the beginning of the buffer region plus an offset when the next destination address is beyond the range of the buffer region.

21. (Original) The computer-readable medium of claim 13 wherein the input process instruction sequence includes a write module instruction sequence that causes the processor to store random bits by minimally causing the processor to:

store a quantity of random bits in the buffer region at a destination address;

advance the destination address to the next location in a different block in the buffer region;

place the destination address to the beginning of the buffer region plus an offset when the next destination address is beyond the range of the buffer region.

22. (Original) The computer-readable medium of claim 13 wherein the input process instruction sequence includes a write module instruction sequence that causes the processor to store random bits by minimally causing the processor to:

store a quantity of random bits in the buffer region at a destination address; and

advance the destination address to the next location in different block in the buffer region

wherein the different block is chosen according to a least recently updated block.

23. (Original) The computer-readable medium of claim 13 wherein the output process instruction sequence includes an increment instruction sequence that causes the processor to alter the random bits at the source location by minimally causing the processor to increment in the buffer region in the memory a digital value representing the retrieved quantity of random bits.

24. (Original) The computer-readable medium of claim 13 wherein the output process instruction sequence includes a transform function that causes the processor to generate a new quantity of random bits by minimally causing the processor to apply a one-way hash function to the retrieved quantity of random bits.

25. (Original) An apparatus for providing random bits comprising:

means for storing random bits in a buffer;

means for retrieving a quantity of random bits from the buffer at a source location;

means for altering the random bits in the buffer at the source location;

means for generating a new quantity of random bits based on the retrieved quantity of random bits;

means for advancing the source location to the next location in the buffer; and

means for placing the source location to the beginning of the buffer plus an offset when the next location is beyond the range of the buffer.

26. (Original) The apparatus of claim 25 wherein the means for storing random bits in

the buffer comprises:

means for storing a quantity of random bits in a buffer at a source location;

means for advancing the destination location to the next destination location in the buffer; and

means for placing the destination location at the beginning of the buffer plus an offset when the next destination location is beyond the limit of the buffer.

27. (Original) The apparatus of claim 25 wherein the means for storing random bits in the buffer comprises:

means for organizing the buffer into blocks;

means for storing a quantity of random bits at a chosen destination location in a first block in the buffer;

means for advancing the destination location to chosen destination location in a different block in the buffer; and

means for placing the destination location to a chosen destination location in a different block at the beginning of the buffer plus an offset when the next destination location is beyond the limit of the buffer.

28. (Original) The apparatus of claim 25 wherein the means for storing random bits in the buffer comprises:

means for organizing the buffer into blocks;

means for storing a quantity of random bits in a first block at a chosen destination location; and

means for advancing the destination location to a destination location in a different chosen block wherein the block is chosen according to a least recently updated block.

29. (Original) The apparatus of claim 25 wherein the means for altering the random bits in the buffer at the source location comprises means for incrementing a digital value comprised of the retrieved quantity of random bits.

30. (Previously presented) The apparatus of claim 25 wherein the means for generating a new quantity of random bits based on the retrieved quantity of random bits comprises a means for applying a one-way hash function to the retrieved quantity of random bits.

31. (Currently amended) A method for providing random bits comprising:

receiving random bits into a buffer;

receiving a request for random bits from a consuming process;

authorizing the consuming process to access the buffer while precluding other processes from accessing the buffer; and

providing random bits from the buffer to the authorized consuming process, wherein providing random bits comprises:

providing random bits from the buffer when the buffer is not empty; and

receiving additional random bits into the buffer when the buffer becomes depleted.

32. (Original) The method of claim 31 wherein receiving random bits into a buffer comprises:

receiving a semaphore;

storing random bits into a buffer; and

relinquishing the semaphore when no other random bits are to be stored in the buffer.

33. (Original) The method of claim 31 wherein authorizing the consuming process comprises:

providing a semaphore to the consuming process when a sufficient quantity of random bits is present in the buffer; and

waiting for the consuming process to return the semaphore.

34. (Original) The method of claim 31 wherein authorizing the consuming process comprises:

providing a semaphore to the consuming process; and

waiting for the consuming process to return the semaphore.

35. (Cancelled)

36. (Original) The method of claim 31 wherein providing random bits comprises:

allowing the consuming process to retrieve random bits from the buffer; and

requiring the consuming process to abate when the buffer is empty.

37. (Original) An apparatus for providing random bits comprising:

buffer;

input manager that is capable of receiving random bits and storing said random bits in the buffer;

arbiter capable of selecting a request from among a plurality of requests for random bits; and

output manager that is capable of providing a quantity of random bits in response to the selected request.

38. (Original) The apparatus of claim 37 wherein input manager comprises a buffer request unit that issues a buffer request signal to an arbiter when random bits are to be stored in the buffer and stores random bits into the buffer when it receives a buffer grant signal.

39. (Original) The apparatus of claim 37 wherein the arbiter accepts a request in the form of a demand quantity indicator and issues a grant signal when a sufficient quantity of random bits are available in the buffer to satisfy the request according to the demand quantity indicator and wherein the output manager retrieves a quantity of random bits from the buffer when the arbiter issues a grant signal.

40. (Original) The apparatus of claim 37 wherein the arbiter issues a single corresponding grant signal in response to a plurality of request signals and maintains the corresponding grant signal as long as a corresponding request signal remains active and wherein the output manager retrieves a quantity of random bits from the buffer when the arbiter issues a grant signal.

41. (Original) The apparatus of claim 37 wherein the input manager stores additional random bits into the buffer when the buffer is depleted.

42. (Original) The apparatus of claim 37 wherein the arbiter de-asserts a grant request

when the buffer is empty.

43. (Currently amended) A random bit provisioning unit comprising:

processor for executing instructions;

memory; and

instruction sequences stored in the memory including:

arbitration process instruction sequence that, when executed by the processor, minimally causes the processor to authorize one consuming process in response to a request for random bits from one or more consuming processes;

input process instruction sequence that, when executed by the processor, minimally causes the processor to store random bits in a buffer region in the memory; and

output process instruction sequence that, when executed by the processor, minimally causes the processor to:

provide random bits to an authorized consuming process while precluding a non-authorized consuming process from access said buffer region,

wherein the output process instruction sequence minimally causes the processor to provide random bits by minimally causing the processor to:

allow an authorized consuming process to retrieve random bits from the buffer region when the buffer region is not empty; and

store additional bits in the buffer region when the buffer region becomes depleted.

44. (Original) The random bit provisioning unit of claim 43 wherein the input process instruction sequence minimally causes the processor to store random bits in a buffer region by

minimally causing the processor to:

receive a semaphore;

store random bits in the buffer region; and

relinquish the semaphore when it is finished storing random bits in the buffer region.

45. (Original) The random bit provisioning unit of claim 43 wherein the arbitration process instruction sequence minimally causes the processor to authorize one consuming process by minimally causing the processor to:

receive a request for random bits that includes a demand quantity indicator;

issue a semaphore to a consuming process in response to the request when a sufficient quantity of random bits is available in the buffer region according to the demand quantity indicator; and

dwell the arbitration process until the semaphore is received back from the consuming process.

46. (Original) The random bit provisioning unit of claim 43 wherein the arbitration process instruction sequence minimally causes the processor to authorize one consuming process by minimally causing the processor to:

receive a request for random bits;

issue a semaphore to a consuming process in response to the request; and

dwell the arbitration process until the semaphore is received back from the consuming process.

47. (Cancelled)

48. (Original) The random bit provisioning unit of claim 43 wherein the output process instruction sequence minimally causes the processor to provide random bits by minimally causing the processor to:

allow an authorized consuming process to retrieve random bits from the buffer region; and

direct the consuming process to relinquish a semaphore back to the arbitration process when the buffer region is empty.

49. (Currently amended) A computer readable medium having computer-executable functions for providing random bits comprising:

arbitration process instruction sequence that, when executed by a processor, minimally causes the processor to authorize one consuming process in response to a request for random bits from one or more consuming processes;

input process instruction sequence that, when executed by a processor, minimally causes the processor to store random bits in a buffer region in a memory; and

output process instruction sequence that, when executed by a processor, minimally causes the processor to:

provide random bits to an authorized consuming process while precluding a non-authorized consuming process from access said buffer region,

wherein the output process instruction sequence minimally causes a processor to provide random bits by minimally causing the processor to:

allow an authorized consuming process to retrieve random bits from a buffer region in a memory when the buffer region is not empty; and

store additional bits in the buffer region when the buffer region becomes depleted.

50. (Original) The computer readable medium of claim 49 wherein the input process instruction sequence minimally causes a processor to store random bits in a buffer region by minimally causing the processor to:

receive a semaphore;

store random bits in a buffer region in a memory; and

relinquish the semaphore when it is finished storing random bits in the buffer region.

51. (Original) The computer readable medium of claim 49 wherein the arbitration process instruction sequence minimally causes a processor to authorize one consuming process by minimally causing the processor to:

receive a request for random bits that includes a demand quantity indicator;

issue a semaphore to a consuming process in response to the request when a sufficient quantity of random bits is available in a buffer region in a memory according to the demand quantity indicator; and

dwell the arbitration process until the semaphore is received back from the consuming process.

52. (Original) The computer readable medium of claim 49 wherein the arbitration process instruction sequence minimally causes a processor to authorize one consuming process

by minimally causing the processor to:

receive a request for random bits;

issue a semaphore to a consuming process in response to the request; and

dwell the arbitration process until the semaphore is received back from the consuming process.

53. (Cancelled)

54. (Original) The computer readable medium unit of claim 49 wherein the output process instruction sequence minimally causes a processor to provide random bits by minimally causing the processor to:

allow an authorized consuming process to retrieve random bits from the buffer region; and

direct the consuming process to relinquish a semaphore back to the arbitration process when the buffer is empty.

55. (Original) An apparatus for providing random bits comprising:

means for receiving random bits;

means for receiving a request for random bits;

means for authorizing a consumer of random bits to receive the received random bits while preventing other consumers from accessing the random bits; and

means for providing random bits to an authorized consumer,

wherein the means for providing random bits comprises:

means for providing random bits from a buffer when the buffer is not empty; and

means for receiving additional random bits into the buffer when the buffer is depleted.

56. (Original) The apparatus of claim 55 wherein the means for receiving random bits comprises:

means for receiving a semaphore;

means for storing random bits in a buffer; and

means for relinquishing the semaphore when no additional random bits are to be stored in the buffer.

57. (Original) The apparatus of claim 55 wherein the means for authorizing a consumer comprises:

means for providing a semaphore to a consuming process when a sufficient quantity of random bits is available to satisfy the request for random bits; and

means for dwelling an authorization process until the consuming process returns the semaphore.

58. (Original) The apparatus of claim 55 wherein the means for authorizing a consumer comprises:

means for providing a semaphore to a consuming process; and

means for dwelling an authorization process until the consuming process returns the semaphore.

59. (Cancelled)

60. (Original) The apparatus of claim 55 wherein the means for providing random bits comprises:

means for allowing a consuming process to receive random bits from the buffer; and

means for requiring the consuming process to relinquish a semaphore when the buffer is empty.